Правительство Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Санкт-Петербургский государственный университет»

Кафедра информационно-аналитических систем

Воронин Юрий Игоревич

Система электронной подписи

Бакалаврская работа

Научный руководитель:
д. ф.-м. н., профессор Крук Е. А.

Рецензент:
д. т. н. Татарникова Т. М.

Санкт-Петербург
2017

SAINT-PETERSBURG STATE UNIVERSITY

Chair of The Analytical Information Systems

Yury Voronin

Digital signature system

Bachelor's Thesis

Scientific supervisor:
professor Eugeny Krouk

Reviewer:
professor Tatiana Tatarnikova

Saint-Petersburg
2017

Оглавление

Введение

В настоящее время, в связи увеличением документооборотов между различными компаниями и физическими лицами, активно развивается обмен электронными версиями этих самых документов, а не их оригиналами. Это, в первую очередь, уменьшает затраты на бумагу и облегчает хранение больших объемов документов. И, во-вторых, скорость обмена электронными версиями данных на порядок больше скорости обмена бумажными документами, к тому же уменьшаются затраты на передачу. Значит ли это, что обмен электроными копиями документов по всем статьям лучше обмена их бумажными оригиналами?

Нет, и на то есть множество причин, основными из которых являются возможность перехвата сообщения, подмена его содержимого, а также возможность отправителя отказаться от авторства данного документа ввиду отсутствия доказательств обратного. Все это мешает передаче электронных документов называться безопасной. Как же можно избежать этих проблем?

Существует множество различных вариантов, начиная от создания защищенного канала между адресатами, что очень невыгодно в финансовом плане, и заканчивая использованием открытых каналов связи, но с попытками скрыть сам факт передачи данных. Таким скрытием занимается наука стенография.

Но в данной работе речь пойдет только об одном методе, обеспечивающем безопасную передачу данных по сети, а именно передачи зашифрованных сообщений по открытым каналам связи, расшифровать которые могут только участники этого обмена [20]. Изучением такого процесса, еще называемого тайной передачей, занимается наука криптография. Наука, которая, наоборот, изучает методы перехвата сообщений, называется криптоанализом. В данной работе он будет рассмотрен в меньшей степени.

Основной идеей криптографии является передача зашифрованных данных, таким образом, чтобы только принимающая сторона имела возможность их расшифровки. Существует две основных разновидно-

сти криптосистем [21]. Первая из них, называемая криптосистемой с симметричным ключом, подразумевает использование обеими сторонами одинаковых ключей, известных только участникам обмена, для шифрования и дешифрования сообщения. Плюсами таких систем является простота их реализации и скорость шифрования и дешифрования данных. Но, с другой стороны, для правильного функционирования такого рода систем необходимо обеспечить огромное количество ключей в широких сетях для всевозможных пар участников обмена, чтобы эти самые ключи не были одинаковыми даже для двух разных обменов, а также наладить между всеми сторонами секретную передачу ключей (возможно с использованием засекреченных каналов, что связано с немалыми затратами). Поэтому, в связи с вышеописанными сложностями, в настоящее время криптосистемы с открытым ключом почти не используются на практике.

Второй, более популярный в настоящее время, тип криптосистем называется криптосистемы с открытым ключом (или асимметричные криптосистемы). Он основывается на идее использования так называемой функции-лазейки , то-есть функции, обратную к которой для произвольного аргумента найти невозможно (или это вляется NP-трудной задачей), но существует дополнительный параметр, с помощью которого можно обратить эту функцию. Таким образом, открытым ключом является функция, с помощью которой осуществлялось шифрование, а секретным ключом - параметр, с помощью которого эту самую функцию можно обратить. Здесь шифрование сообщения происходит с помощью секретного ключа, а дешифрование – с помощью открытого. Таким образом, решается проблема передачи всем сторонам обмена различных секретных ключей, заменяя их одним открытым.

Однако, что если, несмотря на все старания, наше сообщение удалось перехватить, расшифровать и подделать содержимое. Как узнать, что в пришедшем сообщении нет лишней информации, и автор сообщения – именното лицо, что его отправило? Для решения такой проблемы в криптографии используются схемы электронной подписи, являющиеся основным объектом рассмотрения данной работы. Электронные под-

писи являются аналогами обычных подписей документов и служат для идентификации автора и проверки целостности и правдивости содержимого передаваемого сообщения. Таким образом, по правильно построенной подписи можно доказать авторство сообщения, даже если сам автор отказывается от него.

По используемым алгоритмам при построении подписи все криптосистемы можно условно разделить на уязвимые к атакам с использованием кванотового компьютера и постквановые. К первым относятся криптосистемы, использующие известные NP-трудные задачи при построении подписи (задача факторизации и задача дискретного логарифмирования), тем самым делая себя уязвимыми перед атаками квантовым компьютером, выполняющим гораздо большее число операций, по сравнению с современными компьютерами. Основными представителями таких криптосистем являются система RSA и система Эль-Гамаля. Ко второму же типу относятся те, в которых при построении подписи используются либо проблемы, не имеющие даже неполиномиальных алгоритмов разрешения, либо использующие одноразовые подписи на основе случайных чисел. Здесь можно выделить системы McEliece (и ее раширение – систему Нидеррайтера), опирающиеся на коды исправления ошибок и систему Лампорта, основанную на стойкости hash-функций.

Данная работа посвящена изучению различных криптосистем и способу построения электронных подписей в них, а также атак, проводимых на данные системы.

1. Постановка задачи

В данной работе было решено реализовать одну из постквантовых криптосистем, а именно криптосистему Лампорта, устранить ее одноразовость, а также перейти к использованию устойчивых к коллизиям hash-функций в данной системе. Для этого были поставлены следующие подзадачи:

- Подробно изучить криптосистемы, неотносящиеся и относящиеся к постквантовым;
- Изучить атаки, совершаемые на криптосистемы различных типов;
- Реализовать систему Лампорта с использованием стандартных hash-функций из библиотеки java.security;
- Устранить одноразовость такой подписи с помощью деревьев Меркла;
- Реализовать устойчивую к коллизиям hash-функцию на основе решеток.

2. Непостквантовые криптосистемы

Как уже говорилось ранее, в криптосистемах, не относящихся к постквантовым, для построения и проверки подписи используются известные NP-трудные задачи. Первой такой задачей является задара факторизации числа (разложения числа на простые множители). Основной криптосистемой, использующей задачу факторизации при шифровании и генерации электронной подписи, является система RSA. В настоящее время в данной системе используются числа порядка 2^{1024} для факторизации, так как существуют эффективные алгоритмы разложения для чисел порядка 2^{128} и 2^{256}.

Вторая основная задача – задача дискреного логарифмирования, заключающаяся в обращении уравнения следующего вида

$$b^x \equiv a(modp) \tag{1}$$

Здесь a, b и p – заранее известны, а x, которое должно быть целым и неотрицательным, называется дискретным логарифмом a по основанию p. В качестве мультипликативной группы обычно используется кольцо вычетов по модулю p. p рекомендуется брать в пределах 1024. Основной криптосистемой, использующей эту задачу, является система Эль-Гамаля.

Рассмотрим подробнее каждую из этих систем, построение электронных подписей в них и различные типы атак, используемые при взломе представленных криптосистем.

2.1. Система RSA

Данная система была впервые представлена в 1977 году тремя математиками Рональдом Риверсом, Ади Шамиром и Леонардом Адлеманом, по первым буквам фамилий которых и была названа [8].

2.1.1. Алгоритм работы

На первом этапе создания электронной подписи с помощью алгоритма RSA генерируются открытый и секретный ключи по следующему алгоритму. Сначала выбираются два больших простых числа p и q и вычисляется их произведение $n = p*q$, После чего, с помощью функции Эйлера для числа n ($\phi(n) = (p-1)*(q-1)$), вычисляются секретная и открытая экспоненты, являющиеся частями сектретного и открытого ключей соответственно.

Для вычисления открытой экспоненты необходимо выбрать число e, удовлетворяющее неравенствам $1 < e < \phi(n)$ и взаимно простое с $\phi(n)$. Далее мы вычисляем число d, обратное к числу e по модулю $\phi(n)$, то-есть удовлетворяющее уравнению $d*e \equiv 1(mod\phi(n))$. Полученное число d и является секретной экспонентой.

Теперь в качестве открытого ключа мы берем (n, e), а в качестве секретного ключа: (n, d).

Криптосистема RSA позволяет шифровать сообщения, имеющие вид целого числа, большего 0, но меньшего $n-1$. Для него создается подпись с использованием секретного ключа (n, d) по формуле $s = m^d(modn)$. В результате, вместо нашего сообщения m, передается сообщение с подписью (m, s).

При приеме сообщения с подписью (m, s) для проверки истинности содержания и автора сначала с помощью открытого ключа (n, e) вычисляется прообраз m' по формуле $m' = s^e(modn)$. И во конце истинность проверется путем сравнения m и m'. Именно здесь используется сложность задачи факторизации, а имеено, если приниматель не знает открытую экспоненту, ему придется вычислять значение $\phi(n)$, что требует знания разложения числа n на простые множители.

Обычно для ускорения работы алгоритма шифрования используется маленькая открытая экспонента (то-есть та, в двоичной записи которой мало единичных бит) [11]. Это значительно ускоряет процесс возведения большого числа в степень, но и ослабляет ее, о чем будет рассказано в следующем разделе.

2.1.2. Элементарные атаки на систему RSA

Самые простейшие атаки на криптосистему RSA обычно основываются на реализации конкретных экземпляров данной системы [19].

Так, например, чтобы избежать рассылки всем пользователям своих уникальных значений $n = pq$, можно для всех использовать одно и то же число n. Поэтому, если перехватчик знает n, его разложение на p и q, он может вычислить секретный ключ пользователя с помощью китайской теоремы об остатках. А открытый нам и так известен.

Также можно узнать саму подпись пользователя для сообщения m [3]. Для этого необходимо сгенерировать поддельное сообщение $m' = r^e M(mod n)$, где r – произвольное целое положительное число, меньшее m, а e – часть открытого ключа пользователя, сообщение которого мы хотим взломать. И далее, пользуясь равенством $s = s'/r(mod n)$, где s' – подпись сообщения m', мы можем подделать подпись данного пользователя для данного сообщения.

Второй же возможной особенностью криптосистемы RSA, как было сказано раньше, является использование маленькой закрытой или открытой экспоненты при подписи сообщения (то-есть числа, в двоичной записи которого присутствует малое число единичных битов). С такой особенностью связана атака Винера, основывающаяся на последовательном приближении дробями. Для возможности такой атаки небходимо, чтобы, например, секретный ключ d удовлетворял неравенству $d < \frac{1}{3}n^{1/4}$.

Однако всех этих атак недостаточно, чтобы взломать хорошую криптосистему RSA, использующую большие экспоненты и разные n для всех пользователей, так как их сложность становится экспоненциальной. Однако и их можно взломать, используя алгоритм Шора, разработанный специально для квантовых компьютеров. Его время работы будет уже полиномиальным. В следующих разделах он будет подробно разобран.

2.1.3. Квантовый компьютер

Рассмотрим некоторую физическую систему. Она может находиться в одном из N возможных состояний, называемых $|1\rangle ... |N\rangle$ соответственно. Такие состояния называются классическими. Однако, в квантовой физике рассматриваются так называемые квантовые состояния системы, являющиеся суперпозицией классических

$$|\phi\rangle = \alpha_1|1\rangle + ... + \alpha_N|N\rangle = \sum_{i=1}^{N} \alpha_i|i\rangle \tag{2}$$

В этом уравнении α_i – комплексные числа, удовлетворяющие следующему равенству $\sum\limits_{i=1}^{N} |\alpha_i|^2 = 1$. Они называются амплитудами состояний $|i\rangle$. С такого рода состояниями мы умеем проводить две различные операции, во-первых, измеряять их, а, во-вторых, применять к ним унитарное преобразование.

Измерением состояния системы называется определение сотояния, в котором система находится в данный момент времени. Для классических систем это сделать легко, однако для квантовых мы не сможем со стапроцентной точностью указать такое состояние. Однако, зная вектор $(\alpha_1, ..., \alpha_N)$ мы можем сказать, что система находится в состоянии $|i\rangle$ с вероятностью α_i.

Второй операцией является применение унитарного преобразования к текущему квантовому состоянию. Пусть мы получили новое состояние $|\psi\rangle = \beta_1|1\rangle + ... + \beta_N|N\rangle$. Значит, можно сказать, существует комплекснозначная матрица U, удовлетворяющая уравнению представленному на (рис.1)

$$U \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_N \end{pmatrix}$$

Рис. 1: Унитарное преобразование

Вектор из α и вектор из β являются вероятностными, значит матрица U – унитарная, так как сохраняет норму при преобразовании. Зна-

чит, любое преобразование квантовых состояний является унитарным.

Аналогично квантовой физике, в квантовых компьютерах используются так называемые квантовые биты или кубиты [15]. Если один бит стандартного компьютера может принимать два состояния $|0\rangle$ и $|1\rangle$, то кубит в качестве состояния принимает их суперпозицию $\alpha_0|0\rangle + \alpha_1|1\rangle$, при условии $|\alpha_0|^2 + |\alpha_1|^2 = 1$.

Аналогично описываются состояния для нескольких кубитов. Они представляют из себя набор всевозможных векторов длины n, где n – количество кубитов, состоящих из 0 и 1. И аналогично квантовой физике вводятся операции, которые можно совершать с данными кубитами. Стоит добавить, что измерение можно проводить только по некоторым координатам кубитов, не затрагивая остальные.

Именно такое строение памяти квантового компьютера необходимо для реализации алгоритма Шора, который может взломать почти любую непостквантовую криптосистему за полиномиальное время.

2.1.4. Атака Шора

Сначала надо сказать, что алгоритм Шора не решает напрямую задачу факторизации целого положительного числа на простые множители. Он решает задачу нахождения порядка числа n, а именно наименьшее число $r \geq 1$, являющееся решением уравнения $x^r \equiv 1(modn)$, где x удовлетворяет неравенству $1 < x < n$.

Задача факторизации, в свою очередь, легко сводится к задаче нахождения порядка [18]. Действительно, мы имеем уравнение

$$(x^{r/2} - 1)(x^{r/2} + 1) \equiv 0(modn) \tag{3}$$

Очевидно, что левая скобка не может давать 0 по модулю n, так как r по определению наимешьшее число, такое что $x^r - 1 \equiv 0(modn)$. Значит, имеем $x^{r/2}+1 \equiv 0(modn)$. Значит, если $x \neq 0$, $x \neq 1$ и r – четное, значит $gcd(x^{r/2} + 1, n)$ и есть нетривиальный делитель n.

Важным понятием для алгоритма Шора является дискретное преобразование Фурье порядка q, которое производится с помощью умно-

жения на матрицу следующего вида (рис.2). Здесь $\zeta = e^{2\pi i/q}$.

$$U_q = \frac{1}{\sqrt{q}} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \zeta & \zeta^2 & \cdots & \zeta^{q-1} \\ 1 & \zeta^2 & \zeta^4 & \cdots & \zeta^{2(q-1)} \\ 1 & \zeta^3 & \zeta^6 & \cdots & \zeta^{3(q-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta^{q-1} & \zeta^{2(q-1)} & \cdots & \zeta^{(q-1)^2} \end{pmatrix}$$

Рис. 2: Преобразование Фурье

Такое преобразование необходимо для нахождения периодической структуры последовательности, состоящей из комплексных чисел.

Сам алгоритм Шора, разделен на шесть шагов [9].

На первом шаге, находится число q, удовлетворяющее неравенству $2n^2 < q < 3n^2$ и раскладывающиеся на маленькие простые множители (мы полагаем, что $q = 2^l$). Далее заводится квантовый компьютер с q^2 кубитами основной памяти и инициализируется начальное состояние $|\psi\rangle = |0,0\rangle$. Затем наше состояние приводится к виду, где первые q координат вектора представляют число от 0 до $q-1$: $|\psi\rangle = \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle|0\rangle$. Это делается с помощью блочно-диагональной матрицы, блоки которой имеют вид как на (рис.3)

$$R = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Рис. 3: Блок

Второй шаг представляет из себя применение обратимого преобразования $|a,0\rangle \to |a, x^a (mod n)\rangle$, для произвольного $1 < x < n$.

На третьем шаге производится измерение второй половины кубита для выявления всех таких состояний, что вторая их часть находится в состоянии k, являющегося произвольно выбранным значением

выражения $x^a(modn)$. Значит, новое состояние имеет следующий вид: $|\psi\rangle = \frac{1}{|A|}\sum_{a\in A}|a\rangle|k\rangle$. Очевидно, что все значения a отличаются между собой на r.

Четвертый шаг является применением к нашему состоянию дискретного преобразования Фурье, переводящее его в состояние следующего вида, где $\zeta = e^{2\pi icr/q}$:

$$|\psi\rangle = \sum_{c=0}^{q-1}\frac{e^{2\pi ica_0/q}}{\sqrt{q|A|}}(\sum_{d=0}^{|A|-1}\zeta^d)|c\rangle|k\rangle \tag{4}$$

На пятом шаге мы измеряем произвольное состояние c, появляющееся с вероятностью $\frac{1}{\sqrt{q|A|}}|\sum_{d=0}^{|A|-1}\zeta^d|^2$, где ζ имеет вид как на предыдущем шаге. Далее мы проверяем, насколько близко число $\frac{cr}{q}$ к целому числу d. И если они близки, мы запоминаем число c/q, близкое к d/r. Иначе выбираем другое значение c.

Наконец, на последнем шаге алгоритма работает уже классический компьютер. Мы находим наилучшее приближение числа c/q дробями вида d_i/r_i, знаменатель которых удовлетворяет неравенству $r_i < n$, с помощью метода непрерывных дробей. Для последнего r_i в разложении c/q мы ищем целое число e, такое что при умножении его на r_i выполняется равенство $x^{er_i} \equiv 1(modn)$. Если таких степеней нет – возвращаемся к предыдущему пункту. Иначе считаем значение $y = x^{er_i/2}(modn)$ и находим делитель числа n по формуле $gcd(y+1, n)$.

Таким образом, с помощью этого алгоритма мы можем взломать произвольную криптосистему RSA за полиномиальное время, равное $O((log(n))^2 log(log(n))log(log(log(n))))$, вне зависимости от того, насколько большие n используются в ней для факторизации, или насколько большие в ней открытая и секретная экспоненты.

2.2. Система Эль-Гамаля

Данная криптосистема была предложена Тахером Эль-Гамалем в 1985 году и основывается на NP-трудности задачи дискретного логарифмирования в конечных полях [1].

2.2.1. Алгоритм работы

Как и во всех криптосистемах, в системе Эль-Гамаля на первом этапе генерируются открытый и секретный ключи. Это происходит по следующему алгоритму. Сначала генерируется произвольное простое число p, затем вычисляется число g, являющееся его первообразным корнем (образующим элементом мультипликативной группы кольца вычетов по модулю p). Оно должно удовлетворять уравнению $g^{\phi(p)} \equiv 1(modp)$.

После этого мы должны выбрать произвольное число x, такое что оно является целым и удовлетворяет неравенствам $1 < x < p$. И, наконец, находится число y по формуле $y = g^x(modp)$.

В системе Эль-Гамаля в качесве открытого ключа берется тройка (p, g, y), а в качестве секретного ключа – число x.

С помощью криптосистемы Эль-Гамаля возможно подписать сообщения вида m, где m – произвольное целое неотрицательное число, меньшее p, затем мы выбираем произвольное число k, такое что оно удовлетворяет неравенствам $1 < k < p - 1$ и взаимно просто с $p - 1$. С помощью этого k вычисляется первая часть подписи $r = g^k(modp)$.

Последним шагом мы должны вычислить вторую часть подписи $s = (m - r * x) * k^{-1}(mod(p - 1))$. В итоге, вместо сообщения M передается подписанное сообщение (M, r, s).

Принимающая сторона производит проверку подписи по следующему алгоритму. Сначала проверяется истинность условий $0 < r < p$ и $0 < s < p - 1$. Если хотя бы одно из них неверно - подпись считается недействительной.

После вышеописанной проверки вычисляется значение hash-функции $m = h(M)$ для нашего сообщения M, и происходит проверка подписи на основе проверки равенства $y^r * r^s \equiv g^m(modp)$.

2.2.2. Элементарные атаки на систему Эль-Гамаля

Самые простые способы атак были предложены еще самим Тахером Эль-Гамалем в статье, описывающем его криптосистему [6]. Основная

идея первой атаки заключается в решении системы из l уравнений вида

$$m = xr + ks(mod(p-1)) \tag{5}$$

Здесь m, r и s берутся из набора сообщений с подписями $\{(m_i, r_i, s_i) : i = 1, .., l\}$. Видно, что на самом деле неизвестных тут $l+1$, так как для каждой подписи используется свое k, то решить такую недетерминированную систему непросто, но если у нас есть хотя бы 2 совпадающих между собой k – это становится возможным.

Вторая идея состоит в том, что при попытке подделать подпись сообщения, можно начинать не с r, вычисляемого по формуле $r = a^j(modp)$, ведь тогда при вычислении s потребуется вычисление дискретного логприфма. Вместо этого начать можно с s, и тогда для нахождения r необходимо решить более легкую задачу, чем дискретный логарифм.

Очевидно, что как и для криптосистемы RSA, такие атаки на серьезную систему Эль-Гамаля не приведут к ощутимым результатам. Однако, атака Шора с помощью квантового компьютера может взломать и систему Эль-Гамаля.

2.2.3. Атака Шора

Атака Шора на систему Эль-Гамаля очень похожа на систему RSA, однако требует трех регистров [12]. Она опирается на идею нахождения стабилизатора (порождающего элемента) для следующего преобразования плоскости: $f(z_1, z_2) = g^{z_1}a^{z_2}(modn)$. Тогда мы рассматриваем следующее множество $D = \{(z_1, z_2) : g^{z_1}a^{z_2} \equiv 1(modp)\}$, и, зная его базис, легко определить элементы вида $(k, -1)$, принадлежащие D, и получить уравнение вида $g^k = a$, а значит k и есть искомый дискретный логарифм.

2.3. Выводы

Таким образом, можно сказать, что криптосистемы, не относящиеся к постквантовым, в настоящее время могут повсеместно исполь-

зоваться в связи с низким уровнем производительности современных компьютеров. Для избежания возможности взлома этих криптосистем достаточно просто увеличивать размерность используемых в них NP-трудных задач (длины начальных простых чисел в системе RSA и размерность конечного поля в системах Эль-Гамаля). Однако, сейчас идет активное развитие так называемых квантовых компьютеров, способных выполнять гораздо больше операций в секунду, по сравнению с обычными компьютерами. Тем самым, все используемые в настоящее время криптосистемы системы, не относящиеся к постквантовым, оказываются уязвимыми к атаком с использованием алгоритма Шора и не могут использоваться для шифрования данных и генерации электронных подписей в дальнейшем.

В следующей главе будут рассмотрены системы, устойчивые к атакам с использованием квантового компьютера.

3. Постквантовые криптосистемы

К постквантовым криптосистемам относятся те системы, которые неуязвимы к атакам с использованием квантового компьютера, то-есть те, для которых не существует эффективных алгоритмов атак даже со стороны квантового компьютера, либо системы, использующие одноразовые открытый и секретный ключи и случайные числа для их генерации [4].

К основным типам таких систем относятся

- Системы на основе кодов исправления ошибок (McEliece);
- Системы, основанные на hash-функциях (Лампорт).

Рассмотрим подробнее эти типы криптосистем.

3.1. Система McEliece

Этот алгоритм построения криптосистем был впервые предложен в 1978 году Робертом Мак-Элисом и был первым алгоритмом, использующим случайные числа при шифровании и построении электронных подписей. Как уже говорилось ранее, эти криптосистемы работают на основе кодов исправления ошибок (передача избыточной информации для упреждения ошибок, вызванных помехами при передаче) и матриц случайных чисел.

Но сначала необходимо дать несколько вспомогательной теории для дальнейшего описания работы предтавленной криптосистемы.

3.1.1. Коды исправления ошибок

Линейным кодом C называется k-мерное подпространство n-мерного векторноо пространства над произвольным конечным полем $GF(q)$, где k и n – целые положительные числа, такие что $n \geq k$, а q – степень произвольного простого числа p. Основным показателем линейных кодов является количество ошибок, которое этот код может исправить [14].

Для каждого элемента такого кода задан вес Хэмминга, равный количеству ненулевых компонент вектора x. Он обозначается $\omega(x)$. Также для двух произвольных векторов, принадлежащих одному линейному коду, вводится понятие расстояния Хэмминга, равное числу координат, в которых данные вектора не совпадают.

Порождающей матрицей G для линейного кода C называется матрица, строки которой являются базисными векторами данного k-мерного подпространства, то-есть матрица размера $(k \times n)$, удовлетворяющая следующему уравнению

$$C = \{xG : x \in GF(q)_n^k\} \tag{6}$$

Проверочной же матрицей H, наоборот, называется та матрица, строки которой – базис ортогонального дополнения C, то-есть матрица размера $(n - k \times n)$, удовлетворяющая следующему уравнению

$$C = \{x \in GF(q)_n^k : H^t x = 0\} \tag{7}$$

Говорят, что линейный код C может исправить t ошибок, если $\forall x, y \in C(x \neq y)B(x,t) \cap B(y,t) = \emptyset$, где $B(x,t) = \{y \in \{0,1\}^n : t \geq d(x,y)\}$. Здесь $d(x,y)$ обозначает расстояние Хэмминга между x и y. Основным выводом в теории линейных кодов является то, что линейный код с минимальным расстоянием Хэмминга, равным d, может исправить $\lfloor (d-1)/2 \rfloor$ ошибок. Такие линейные коды называются $[n, k, d]$ линейными кодами.

Еще одним важным определением в теории линейных кодов является определение синдрома. Синдроном вектора v называется вектор, полученный путем умножения матрицы проверки для нашего кода на транспонированный вектор v: Hv^t. Этот вектор помогает при декодировании кода, так как синдром кодового слова, по определению, равен 0, следовательно два слова, различающиеся на кодовое слово, имеют одинаковые синдромы. Значит все вектора в коде можно разбить на классы эквивалентности и выделить в каждом лидера (вектор наименьшего веса Хэмминга). И далее, при приеме слова x строить его синдром, находить соответствующий синдром в таблице, брать лидера класса эк-

вивалентности этого синдрома и, принимая его за ошибку при передаче, вычислять сообщение $c' = x - e_i$.

В классической системе McEliece используются двумерные линейные коды, то-есть $q = 2$, также называемые кодами Гоппа.

3.1.2. Коды Гоппы

Пусть полиномом Гоппа над полем $GF(q)$ называется полином следующего вида

$$g(x) = g_0 + g_1 \cdot x + ... + g_t \cdot x^t = \sum_{i=0}^{n} g_i \cdot x^i \tag{8}$$

Здесь все g_i принадлежат полю $GF(q)$. Пусть L – n-мерное подмножество расширения поля $GF(q)$, такое что $L = \{\alpha_1, ..., \alpha_n\} \subseteq GF(q)$ и $g(\alpha_i) \neq 0$ для любого α_i.

Пусть нам дано кодовое слово $c = (c_1, ...c_n)$ над полем $GF(q)$ и для него мы определяем следующую функцию

$$R_c(x) = \frac{c_i}{x - \alpha_i} \tag{9}$$

Теперь двоичным кодом Гоппа назовем всевозможные кодовые слова, такие что $R_c(x) \equiv 0(modg(x))$ или, другими словами, полином $g(x)$ делит полином $R_c(x)$ [13]. Известно, что размерность k кода Гоппы длины n не меньше, чем $n - mt$, аминимальное расстояние d удовлетво неравенству $d \geq 2t + 1$. Таким образом, в качестве линейного кода исправления ошибок берется двоичный код Гоппы с параметрами $[n, k, d] = [n, \geq n - mt, \geq 2t + 1]$. Бинарные коды Гоппы используются в системе McEliece в связи с возможностью из быстрого декодирования с использованием алгоритма Петерсона.

Заметим, что для кода Гоппы матрица проверки выглядит следующим образом: $H = XYZ$, где строение матриц X, Y и Z представлено на (рис.4)

В заключении стоит сказать, что классические системы McEliece не могут использоваться для построения электронной подписи в свя-

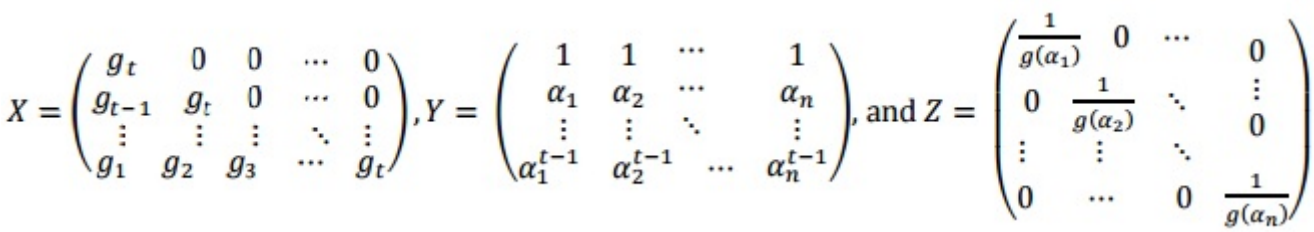

Рис. 4: Матрицы X, Y и Z

зи с тем, что схема подписи не является взаимо-однозначной, однако существует модификация системы McEliece, называемая системой Нидеррайтера. Рассмотрим ее подробнее.

3.1.3. Система Нидеррайтера

Данная система была разработана в 1986 году Харальдом Нидеррайтером, использующая для построения подписи матрицу проверки линейного кода. Однако, только в 2001 году, Куртуа, Финиаз и Сандриер показали, что ее можно использовать для построения электронной подписи [10].

На первом этапе, как и всегда, генерируются открытый и секретный ключи по следующему алгоритму. Сначала выбирается (n, k) код C над полем $GF(q)$, исправляющий t ошибок и обладающий свойством легкого декодирования, и для него строится проверочная матрица H. Затем строятся произвольная невырожденная матрица M над полем $GF(q)$ размера $(n-k \times n-k)$ и произвольная матрица перестановки P размера $(n \times n)$. Открытым ключом системы является пара (H_p, t), где $H_p = MHP$, а закрытым ключом – тройка (S^{-1}, H, P^{-1}).

При отправке сообщения к нему добавляется подпись, строящаяся по следующему алгоритму. Сначала, с помощью hash-функции h вычисляется образ сообщения $s = h(M)$. Отметим, что hash-функция должна возвращать значения длины $n-k$. Затем мы вычисляем последовательно всевозможные hash-значения от аргументов вида $s|i$ и обозначаем их за s_i. Здесь | обозначает конкатенацию строк. Этот процесс останавливается на минимальном значении i_m, таком что s_{i_m}, которое можно рассамтривать как синдром, можно декодировать (обозначим

результат этого декодирования через z). Теперь вместо сообщения M мы отправляем сообщение с подписью (z, i_m).

Для проверки истинности полученного сообщения вида (z, i_m) необходимо сделать следующее. Во-первых, вычислить значение $s_1 = h_p z^T$, а во-вторых, значение $s_2 = h(h(M)|i_m)$. Теперь истинность сообщения и его автора можно проверить путем простого сравнения s_1 и s_2.

Стоит заметить, что оригинальную систему McEliece, основывающуюся на кодах Рида-Соломона, удалось взломать Сидельникову и Шестакову в 1992 году. Сидельников дополнительно смог доказать, что любая атака на систему McEliece может быть полиномиально сведена к атаке на систему Нидеррайтера [17]. Однако модификации данных систем, использующие коды Гоппы все еще остаются криптостойкими. Но возможность взлома одного из кодов исправления ошибок позволило выйти в свет криптосистемам, основывающимся на стойкости hash-функций и одноразовых ключах. Самой распространенной из них является система Лампорта.

3.2. Система Лампорта

Данная система была впервые предложена в 1979 году Лесли Лампортом [7]. Долгое время она считалась плохой, так как сама подпись была в разы больше исходного сообщения, однако, в связи с развитием теории квантовых компьютеров она вновь вернула интерес к себе, так как является кандидатом на то, чтобы быть постквантовой.

В ее основе лежит теория необратимых функций. Функция f называется необратимой, если для произвольного аргумента x вычислить значение $y = f(x)$, но для произвольного значения функции y найти аргумент x, такой что $f(x) = y$, – задача, являющаяся гораздо более трудной, чем поиск значения функции для произвольного аргумента. К сожалению, существование таких функций еще не доказано, но существует множество кандитатов, использующихся в настоящее время в различных криптосистемах.

Стоит добавить, что криптографические hash-функции, помимо необ-

ратимости должны еще обладать устойчивостью как к коллизиям первого рода (для произвольного сообщения M нельзя найти сообщение M', такое что $h(M) = h(M')$), так и к коллизиям второго рода (нельзя найти два различных сообщения с одинаковыми значениями hash-функций). Нарушение этих требований влечет возможность подделки сообщения (например, замена его на другое сообщение, имеющее при этом такое же значени hash-функции).

3.2.1. Алгоритм работы

На этапе генерации открытого и секретного ключей создается n пар случайно сгенерированных чисел, где n – битовый размер выхода для выбранной нами криптографической hash-функции (например, для SHA-1 – 160). Затем все сгенерированные числа преобразуются с помощью имеющейся hash-функции и и навыходе мы имеем массив сгенерированных в качестве секретного ключа и массив их hash-значений в качестве открытого ключа.

Для подписи сообщения необходимо преобразовать наше сообщение в двоичную строку с помощью hash-функции и далее, исходя из того, какой бит (0 или 1) стоит на i-ом месте в данной строке, прибавлять к сообщению первое или втрое число из i-ой пары секретного ключа соответственно.

Для проверки сообщения мы должны сначала вычислить двоичную строку, являющуюся результатом применения hash-функции к самому сообщению и затем, аналогично этапу составления электронной подписи, мы вычисляем значение hash-функции от i-ого элемента подписи и сравниваем его с первым или со вторый двоичным числом из открытого ключа. Если все они совпаают – подпись сообщения признается корректной.

Из плюсов данной системы можно отметить быструю скорость генерации подписи и ее проверки. Из минусов же видна ее одноразовость (так как, подписав хотя бы два сообщения одним и тем же набором значений, мы рискуем полностью выдать наш секретный ключ).

4. Проделанная работа

Как уже было сказано ранее, в качестве практической части работы решено было реализовать одну из разобранных систем. Выбор пал на систему Лампорта в связи с ее возможной постквантовостью и относительной простотой реализации. В качестве среды разработки была выбрана java, в связи с имеющейся в ней библиотекой java.security, в которой реализованы основные криптографические hash-функции MD-5, SHA-1 и SHA-256.

4.1. Реализация системы Лампорта

Была реализована библиотека, включающая в себя применение hash-функции выбранного типа к произвольному сообщению, перевод преизвольного длинного числа из шестнадцатеричной системы счисления в двоичную, генерация подписи Лампорта для произвольного сообщения с использованием секретного ключа, а также проверка полученного подписанного сообщения с помощью открытого ключа.

Результаты работы генератора подписи для сообщения "Hello" приведены на (рис.5)

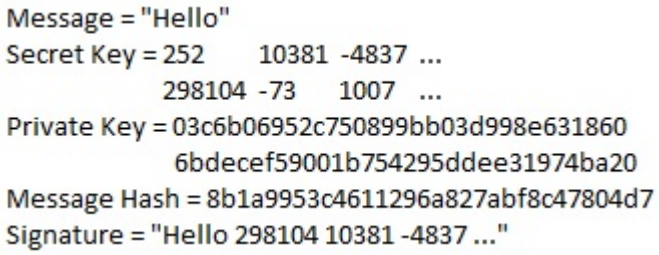

Рис. 5: Подпись Лампорта

Как уже говорилось ранее в нашей библиотеке была реализована возможность генерации hash-значения сообщения с помощью любой доступной в java.security. Это функции MD-5 с 128-битным значением, SHA-1 со 160-битным значением и SHA-256 с 256-битным знаением.

4.2. Реалиация деревьев Меркла

Основным минусом системы Лампорта является ее одноразовость. Этот минус был исправлен с помощью конструкции, называемой деревом Меркла, позволяющей с помощью одного открытого ключа зашифровать до 2^n различных сообщений, где n – высота дерева, которую можно применить к произвольной одноразовой системе электронной подписи [2].

Для реализации дерева Меркла на первом шаге необходимо сгенерировать 2^n пар секретных ключей и соответствующих им открытых ключей, являющимися применением hash-функции к секретным ключам. 2^n открытых ключей становятся листами нашего дерева, а значение для каждой родительской вершины вычисляется по формуле $a_{i,j} = h(a_{i-1,2j}|a_{i-1,2j+1})$, где | обозначает конкатенацию. Строение дерева Меркла для восьми пар секретных (оно имеет высоту 3) и открытых ключей представлено на (рис.6)

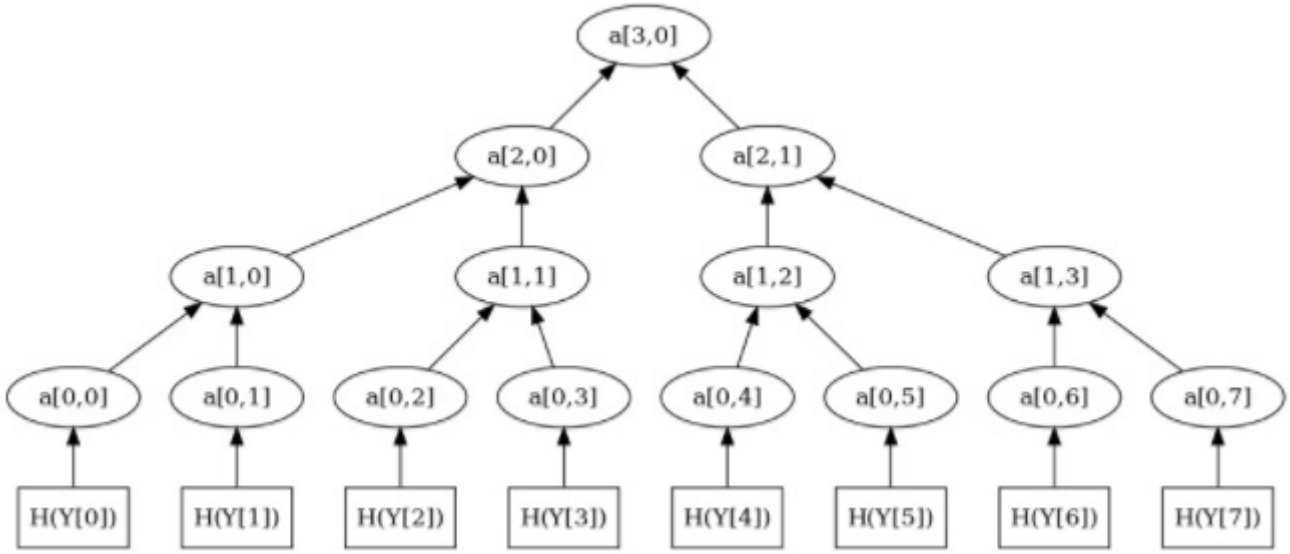

Рис. 6: Дерево Меркла

Теперь мы можем взять в качестве открытого ключа корень p данного дерева. Это дерево позволяет зашифровать 2^n сообщений с помощью открытых ключей, находящихся в листах дерева Меркла. Однако изменится и сам процесс подписи и ее проверки.

Теперь для генерации подписи необходимо выбрать произвольный

ключ i, который ранее не использовался, с помощью него сгенерировать подпись sig', а затем добавить к этой подписи всех братьев по пути от листа с номером i до корня. В результате вместо сообщения M мы отправляем подписанное сообщение $(M, sig', auth_0, ..., auth_{n-1})$. Пример подобных действий представлен на (рис.7)

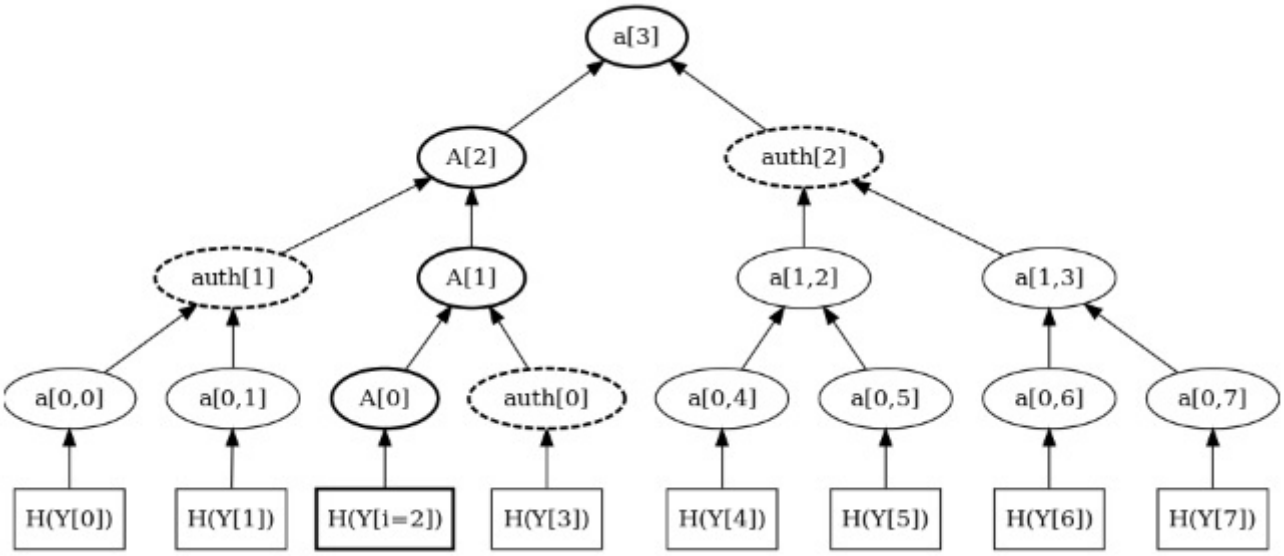

Рис. 7: Подпись Меркла

Здесь жирным выделен выбранный нами ключ, а пунктиром – братья на пути от нашего листа до корня дерева.

При получении подписанного сообщения для установления его истинности необходимо выполнить следующие шаги. Сначала проверить подпись sig', и если она верна, то необходимо последовательно прохешировать пришедших братьев. Для этого сначала мы вычисляем $a_0 = h(x_i)$, где x_i – выбранный секретный ключ.Затем мы вычисляем $a_i = h(a_{i-1}|auth_{i-1})$. Если получившееся a_n совпадает с p, то подпись признается действительной.

Очевидно, что с уничтожением одноразовости подписи Лампорта, мы приобрели новые проблемы, а именно более медленную генерацию подписи и ее проверку.

В нашу библиотеку были добавлены функции, позволяющие получить 2^n пар секретных и открытых ключей, построить для них дерево Меркла, сгенерировать подпись Меркла для произвольного сообщения

и проверить данное подписанное сообщение на корректность.

Пример работы програмыы, строящей дерево и подпись Меркла высотой 1 (позволяющей подписать два различных сообщения) для сообщения "Hello", представлен на (рис. 8)

```
Message = "Hello"
Hash = 8b1a9953c4611296a827abf8c47804d7
Secret Key 1 = 17       8372  -42103 ...
               100128 -4     -129    ...
Public Key 1 = 70efdf2ec9b086079795c442636b55fb  ...
               83fddfd0b861c865446ab1f4bba362d6  ...
Secret Key 2 = -87034  5          11  ...
               21       428111  -4   ...
Public Key 2 = 61be6fe1d065ced6e78516bf37caf58c  ...
               3c59dc048e8850243be8079a5c74d079  ...
p (p=h(PK1||PK2)) = 0c746c6de7f230595520fe20d68d78a1  ...
                    ecb0280da30387bf0d55a2a191e07fcb  ...
i = 1
sig = "Hello" ||
      10128 8372 -42103 ... ||
      0c746c6de7f230595520fe20d68d78a1  ...
      ecb0280da30387bf0d55a2a191e07fcb  ...
```

Рис. 8: Пример работы

4.3. Реализация hash-функций на основе решеток

Как говорилось ранее, стойкойсть системы Лампорта базируется на стойкости hash-функции, используемой в ней. Однако, для всех представленыых в библиотеке java.security hash-функций, а именно MD-5, SHA-1 и SHA-256, удалось доказать существование коллизий и даже привести алгоритмы их поиска. Тем самым использование таких функций ставит под угрозу стойкость криптосистемы, поэтому решено было реализовать устойчивую к коллизиям hash-функцию на основе решеток [16].

Рассмотрин набор линейно-независимых векторов $(b_1, ..., b_n)$ в пространстве R^n. Решеткой, построенной на этих векторах называется множество их линейных комбинаций с целыми коэффициентами

$$L(b_1, ..., b_n) = \{\sum_{i=1}^{n} \alpha_i b_i : \alpha_i \in Z for 1 \leq i \leq n\} \tag{10}$$

Пример решетки в Z^2, построенной на базисе (b_1, b_2) представлен на (рис.9)

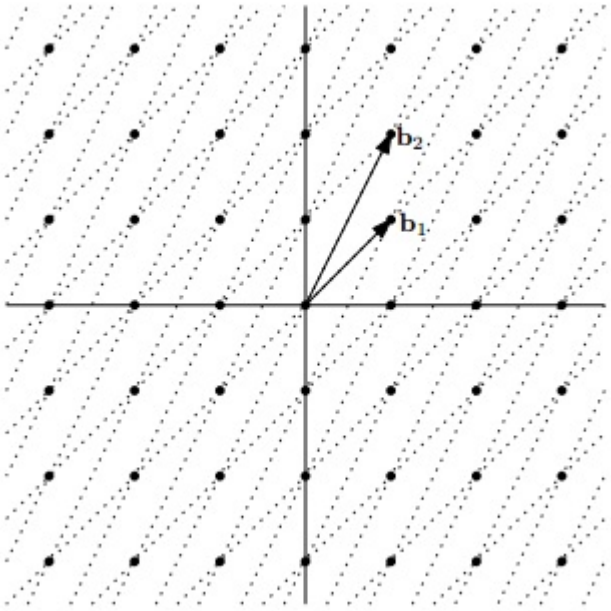

Рис. 9: Решетка в Z^2

Также важными разновидностями решеток являются q-мерные решетки, определяемые для произвольной матрицы A по следующим формулам

$$\Lambda_q(A) = \{y \in Z^m : y = A^T s(modq) forsomez \in Z\} \tag{11}$$

$$\Lambda_q^{\perp}(A) = \{y \in Z^m : Ay = 0(modq)\} \tag{12}$$

Первая решетка построена на базисе из столбцов A, а вторая состоит из векторов, ортогональных столбцам матрицы A.

Определителем решетки называется абсолютное значение определителя матрицы, стоблцы которой есть вектора $b_1, ..., b_n$: $det(L(b_1, ..., b_n)) = |det(B)|$.

Основными вычислительно сложными задачами в теории решеток считаются:

1. Поиск кратчайшего вектора, принадлежащего решетке;

2. Поиск ближайшего вектора, принадлежащего решетке, для произвольного вектора;

3. Поиск минимального по длине набора из n независимых векторов.

Hash-функция, базирующаяся на решетках опирается на первую и вторую из перечисленных проблем [5]. Такая функция зависит от параметров d (модуль пространства входного сообщения), q (модуль пространства результата), n (длина результата), m (длина сообщения). Для того чтобы получить дайджест (значение hash-функции) для произвольного вектора из Z_d^m для произвольного (желательно простого) числа d, необходимо взять матрицу A из $Z_q^{n \times m}$. Тогда $f_A(y) = Ay(modq)$. Очевидно, что такая функция устойчива к коллизиям: как только мы хотим найти два различных вектора y b y', таких что $f_A(y) = f_A(y')$, мы сразу же приходим к проблеме поиска короткого вектора $y - y'$, а если мы знаем один из этих векторов (перехватили сообщение), то к проблеме поиска ближайшего вектора, полиномиального решения для которых не придумано даже для квантового компьютерf.

В нащей библиотеке присутствуют функции, генерирующие произвольную матрицу A, а также считающую hash-значение для любого сообщения, представленного с помощью таблицы ASCII. Для этого полагается следующее: $d = 256$, $q = 1024$, $n = 16$ и в качестве m берется длина входного сообщения.

Рассмотрим сообщение "Hello", представляемое в виде вектора из ASCII кодов символов

$$72, 101, 108, 108, 111 \tag{13}$$

Строение матрицы A представлено на (рис.10). Результатом работы hash-функции является вектор

$$829, 364, 397, 65, 529, 492, 807, 950, 838, 238, 822, 173, 805, 111, 62, 870 \tag{14}$$

7	144	898	2	11
247	994	1001	4	18
743	1	56	0	112
232	764	709	23	11
600	980	1021	78	7
40	36	12	156	1000
9	777	29	341	678
18	923	556	672	1
7	500	0	345	98
180	3	26	5	37
999	0	15	47	74
456	11	80	583	702
542	100	4	673	299
450	1	1004	734	222
807	86	39	2	180
17	592	834	16	1002

Рис. 10: Матрица A

Далее он преобразуется в битовую строку из 160 символов и может быть использован для генерации подписей Лампорта или Меркла.

Также, если дополнительно выполняется условие, что m кратно n можно использовать специальный вид матрицы A для построения hash-функции, обеспечивающий более высокую скорость работы, однако снижающий ее стойкость.

Такая матрица A имеет блочно-диагональный вид, где блоки имеют вид матрицы, стлобцы которой являются циклическими перестановками ее первого столбца. А именно

$$A = [A^{(1)}|...|A^{(m/n)}] \tag{15}$$

Здесь каждая матрица $A^{(i)}$ имеет вид как на (рис.11). Очевидно, что теперь, вместо всей матрицы A, необходимо хранить только порождающие столбцы для матриц $A^{(i)}$

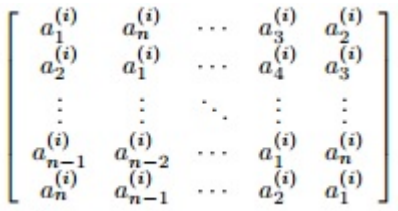

Рис. 11: Строение $A^{(i)}$

В качестве библиотеки для реализации операций над матрицами и векторами была выбрана библиотека Jama.

В нашей библиотеке присутствует и функция, позволяющая получать hash-значение произвольного сообщения с помощью циклических матриц.

5. Заключение

В данной работе были подробно изучены криптосистемы, не относящиеся к постквантовым,а также атаки с использованием квантового компьютера на эти системы.

Также были рассмотрены системы. являющиеся кандидатами на постквантовость, их преимущества и недостатке.

И, наконец, в практической части работы была реализована библиотека, позволяющая строить и проверять подписи Лампорта для произвольных сообщений, базирующаяся на различных hash-функциях, в частности на стандартные MD-5, SHA-1 и SHA-256, а также не реализованные в библиотеке java.security функции, базирующиеся на решетках.

Список литературы

[1] Allen Bryce D. Implementing several attacks on plain ElGamal encryption. — 2008. — URL: http://lib.dr.iastate.edu/cgi/viewcontent.cgi?article=2577&context=etd.

[2] Becker Georg. Merkle Signature Schemes, Merkle Trees and Their Cryptanalysis. — 2008. — URL: https://www.emsec.rub.de/media/crypto/attachments/files/2011/04/becker_1.pdf.

[3] Boneh Dan. Twenty Years of Attacks on the RSA Cryptosystem. — URL: https://crypto.stanford.edu/~dabo/papers/RSA-survey.pdf.

[4] Daniel J. Bernstein Johannes Buchmann Erik Dahmen. Post-Quantum Cryptography. — URL: https://www.politicalavenue.com/libraryebooks/cryptology-and-cryptography/Post%20Quantum%20Cryptography.pdf.

[5] Daniele Micciancio Oded Regev. Lattice-based Cryptography. — 2008. — URL: https://www.cims.nyu.edu/~regev/papers/pqc.pdf.

[6] ElGamal Taher. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. — 1985. — URL: http://caislab.kaist.ac.kr/lecture/2010/spring/cs548/basic/B02.pdf.

[7] Lamport Leslie. Constructing Digital Signatures from One Way Function. — 1979. — URL: https://www.microsoft.com/en-us/research/wp-content/uploads/2016/12/Constructing-Digital-Signatures-from-a-One-Way-Function.pdf.

[8] Milanov Evgeny. The RSA Algorithm. — 2009. — URL: https://sites.math.washington.edu/~morrow/336_09/papers/Yevgeny.pdf.

[9] Moorhouse G. Eric. Shor's Algorithm for Factorizing Large Integers. — URL: https://www.uwyo.edu/moorhouse/slides/talk2.pdf.

[10] Nicolas Courtois Matthieu Finiasz, Sendrier Nicolas. How to achieve a McEliece-based Digital Signature Scheme. — URL: https://www.iacr.org/archive/asiacrypt2001/22480158.pdf.

[11] R.L. Rivest A. Shamir, Adleman L. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. — URL: http://people.csail.mit.edu/rivest/Rsapaper.pdf.

[12] Shor Peter W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. — 1996. — URL: https://arxiv.org/pdf/quant-ph/9508027.pdf.

[13] Valentijn Ashley. Goppa Codes and Their Use in the McEliece Cryptosystems. — 2015. — URL: http://surface.syr.edu/cgi/viewcontent.cgi?article=1846&context=honors_capstone.

[14] Veron Pascal. Code based cryptography and steganography. — 2013. — URL: https://hal.inria.fr/hal-00828034/document.

[15] de Wolf Ronald. Quantum Computing: Lecture Notes. — URL: http://homepages.cwi.nl/~rdewolf/qcnotes.pdf.

[16] van de Pol J.H. Lattice-based cryptography. — 2011. — URL: http://www.cs.bris.ac.uk/pgrad/csjhvdp/files/ThesisJvdPol.pdf.

[17] В. М. Сидельников С. О. Шестаков. О системе шифрования, построенной на основе обобщенных кодов Рида–Соломона. — 1992. — URL: http://www.mathnet.ru/links/03dadf62422f0bbca457acec998579df/dm747.pdf.

[18] Кижватов А.Ю. Богданов и И.С. Квантовые алгоритмы и их влияние на безопасность современных классических криптографических систем. — 2005. — URL: http://crypto.rsuh.ru/papers/bogdanov-kizhvatov-quantum.pdf.

[19] Смарт М. Криптография / Под ред. Ландо. — М. : Техносфера, 2005.

[20] Ященко В.В. Основы криптологии / Под ред. Ященко. — М. : МЦНМО, 2012.

[21] ван Тилборг Х.К.А. Основы криптологии / Под ред. Коряков. — М. : Мир, 2006.